4 Connect Four

Team Members: Talha Dijibril, Aidan Graves, Zach Norton, Ethan Perez

Introduction:

4 Connect Four is our own daring version of the popular game, Connect Four. The game has two players competing on an eight by eight board where they can drop pieces into the desired slots, each player is aiming to connect four of their pieces in a row in any direction. The piece will land at the bottom of the board of the column the user so choses. The users will be updated with the current maximum number of game pieces they have connected. Neither player wins if the board is filled without the required amount of connected pieces.

Requirements:

Rules of Game:

The goal is to have this game highly customizable (the user can set the number of players to 4 max, set the grid to a max size of 16, and set the winning amount of pieces from 2 to the size of the grid). Each player has a different color (red, blue, green, yellow). When it is the player's turn, the player selects a column to place their piece, then the game checks to see if that piece gives them a "matched pieces" count higher than before the piece was placed before moving on to the next player. If the "matched pieces" count is equal to the necessary amount of pieces to win, then the player who reached that number of pieces wins, gains a point, and the game can be restarted.

User Interface:

There are buttons at the top of the grid, each corresponding to a column that, when pressed, will place the piece of the current player. Player 1 has a red 'O', Player 2 has a blue 'X', Player 3 has an orange 'Y', and Player 4 has a purple 'Z'. On the bottom of the screen, the score and max connected pieces of each player is displayed.
If the game is won by any player, a message "Game has been won! Press Yes to replay!" is displayed. If the game resulted in a tie (the board got filled up), the message "Game ended with a tie! Press Yes to replay!" is displayed. Regardless of the message, pressing "Yes" will restart the game. Pressing "No" or "Cancel" will display a message saying "Player 'X' wins with a score of '#'!", where X represents the number of the player with the highest score and # represents their score.

Compiling Code:

```
$ javac -d bin -cp bin src/*.java
```

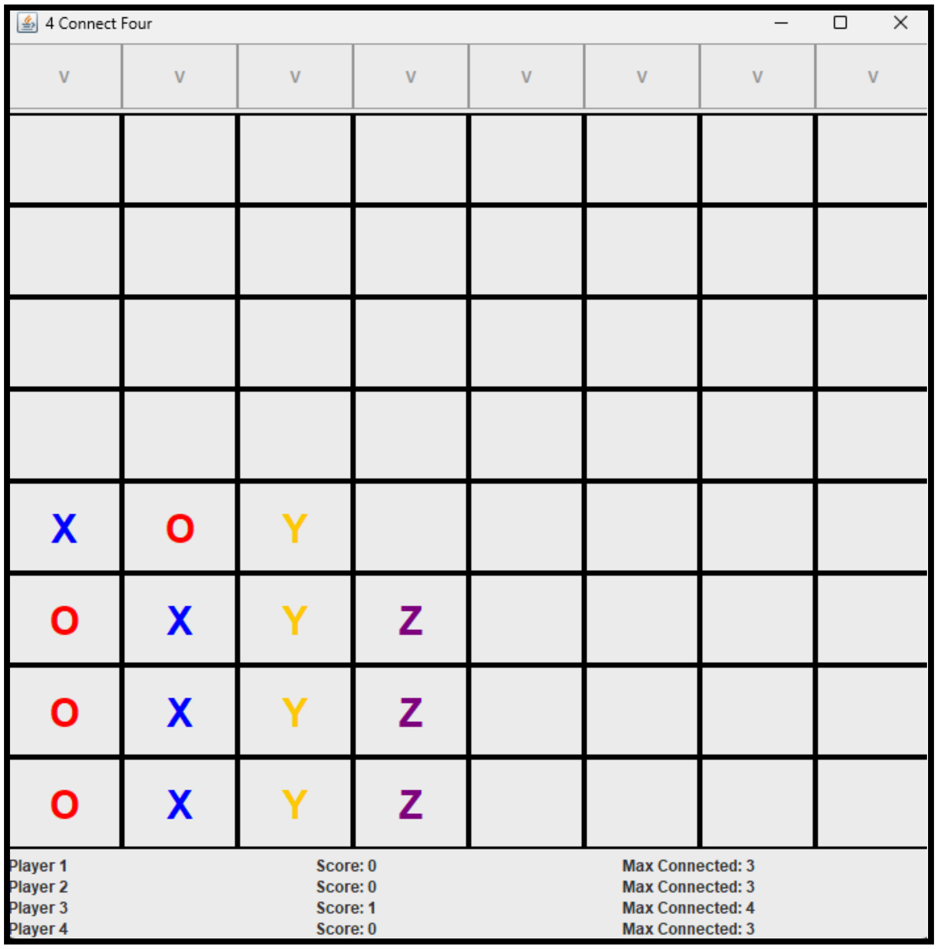
Program Execution:

```
$ java -cp bin ConnectGUI int gridSize int numPlayers int winConnectedPieces
```

Input/Error Handling:

If gridSize is less than 2 or greater than 16, an error message detailing the bounds is displayed.
If numPlayers is less than 1 or greater than 4, an error message detailing the bounds is displayed.
If winConnectedPieces (number of pieces needed to be aligned to win) are less than 1 or greater than the gridSize, an error message detailing the bounds is displayed.
If any of these messages are displayed, the program does not launch.

User Interface Examples:



```
Player 1                 Score: 0                 Max Connected: 3
Player 2                 Score: 0                 Max Connected: 3
Player 3                 Score: 1                 Max Connected: 4
Player 4                 Score: 0                 Max Connected: 3
```

Design:

Project Structure:

- bin
  - Cell$Status.class
  - Cell.class
  - Connect.class
  - ConnectGUI.class
  - Grid.class
  - Player.class
- project_docs
  - SystemTestPlan.pdf
  - ProjectSpecification.pdf
- src
  - Cell.java
  - Connect.java
  - ConnectGUI.java
  - Grid.java
  - Player.java

Classes Outline:

- Cell
  - Instance variables
    - enum Status
    - Status status
  - Constructors
    - Cell()
  - Methods
    - Status getStatus()
    - void setStatus(Status s)
    - String toString()
- Grid
  - Class constants
    - final int MAX_GRID_SIZE = 16;
  - Instance variables
    - int size;
    - Cell[][] grid
  - Constructors
    - Grid(int size)
  - Methods
    - int getSize()
    - Cell[][] getGrid()

        - Cell.Status getStatus()
        - void updateStatus(Cell.Status s)
        - boolean equals(Object o)
        - String toString()
- Player
    - Instance variables
        - int maxConnected
        - int score
    - Constructors
        - Player()
    - Methods
        - int getMaxConnected()
        - int getScore()
        - void setMaxConnected(int numPieces);
        - void increaseScore()
        - boolean equals(Object o)
        - String toString()
- Connect
    - Class constants
        - final int MIN_PLAYERS = 2
        - final int MAX_PLAYERS = 4
        - final int MAX_GRID_SIZE = 16
    - Instance variables
        - int turnNum
        - int numPlayers
        - int gridSize
        - int maxTurn
        - int winConnectedPieces
        - boolean isGameOverWithWinner
        - boolean isGameOverTie
        - Player[] players
        - Grid grid
        - Player currentPlayer
    - Constructor
        - Connect(int gridSize, int numPlayers, int maxConnectedPieces)
    - Methods
        - void determineCurrentPlayer(int turnNum)
        - Cell.Status determineStatus(Player currentPlayer)
        - int placePiece(int col)
        - int findMatchingPieces(int row, int col, Cell.Status pieceStatus)
        - int findHorizontalConnectedPieces(int row, int col, Cell.Status pieceStatus)
        - int findDownConnectedPieces(int row, int col, Cell.Status pieceStatus)
        - int findLeftDiagonalConnectedPieces(int row, int col, Cell.Status pieceStatus)
        - int findRightDiagonalConnectedPieces(int row, int col, Cell.Status pieceStatus)
        - boolean getIsGameOverWithWinner()

        - boolean getIsGameOverTie()
        - Grid getGrid()
        - Player getCurrentPlayer()
        - Player getPlayerAtIndex(int index)
        - void replay()
- ConnectGUI
    - Class constants
        - final Font WORD = new Font(“Courier”, 1, 30)
        - final int DEFAULT_WINDOW_SIZE = 720
        - final int DEFAULT_GRID_SIZE = 8
        - final int DEFAULT_NUM_PLAYERS = 2
        - final int DEFAULT_WIN_CONNECTED_PIECES = 4
        - final Border CELL_BORDER = new LineBorder(Color.BLACK, 2)
        - final Color PURPLE = new Color(128, 0, 128)
        - final int DEFAULT_WINDOW_HEIGHT = 50
        - final int WINDOW_HEIGHT = 500
        - final int MAX_GRID_SIZE = 16
        - final int MAX_PLAYERS = 4
    - Instance variables
        - JPanel buttonPanel
        - JPanel gridPanel
        - JPanel scoreboardPanel
        - Connect connect
        - JLabel[][] cellLabels
        - JButton[] colButtons
        - JLabel[][] playerInfo
        - int gridSize
        - int numPlayers
    - Constructor
        - Connect connect(int gridSize, int numPlayers, int winConnectedPieces)
    - Methods
        - static void main(String[] args)
        - void actionPerformed(ActionEvent e)